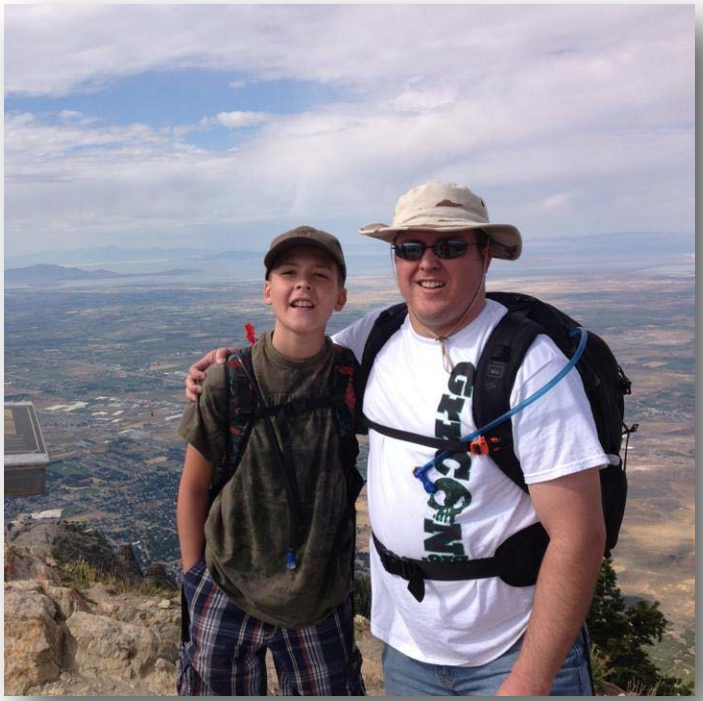
Analyzing Web Attacks Using ModSecurity

Jason Wood
Sr. Security Consultant
Secure Ideas

Background Info

- Work for Secure Ideas
- Penetration Tester, Security Engineer & Systems Administrator
  - Web environments for 11 years
  - Infrastructure
  - Windows, Linux, UNIX
  - Went to the dark side in 2006
- CyberPatriot Mentor
- Family Man



Outline

- ModSecurity 101
- Audit Logging
- Preparation and Configuration
- Identifying Attackers
- Custom Rules
- Analysis

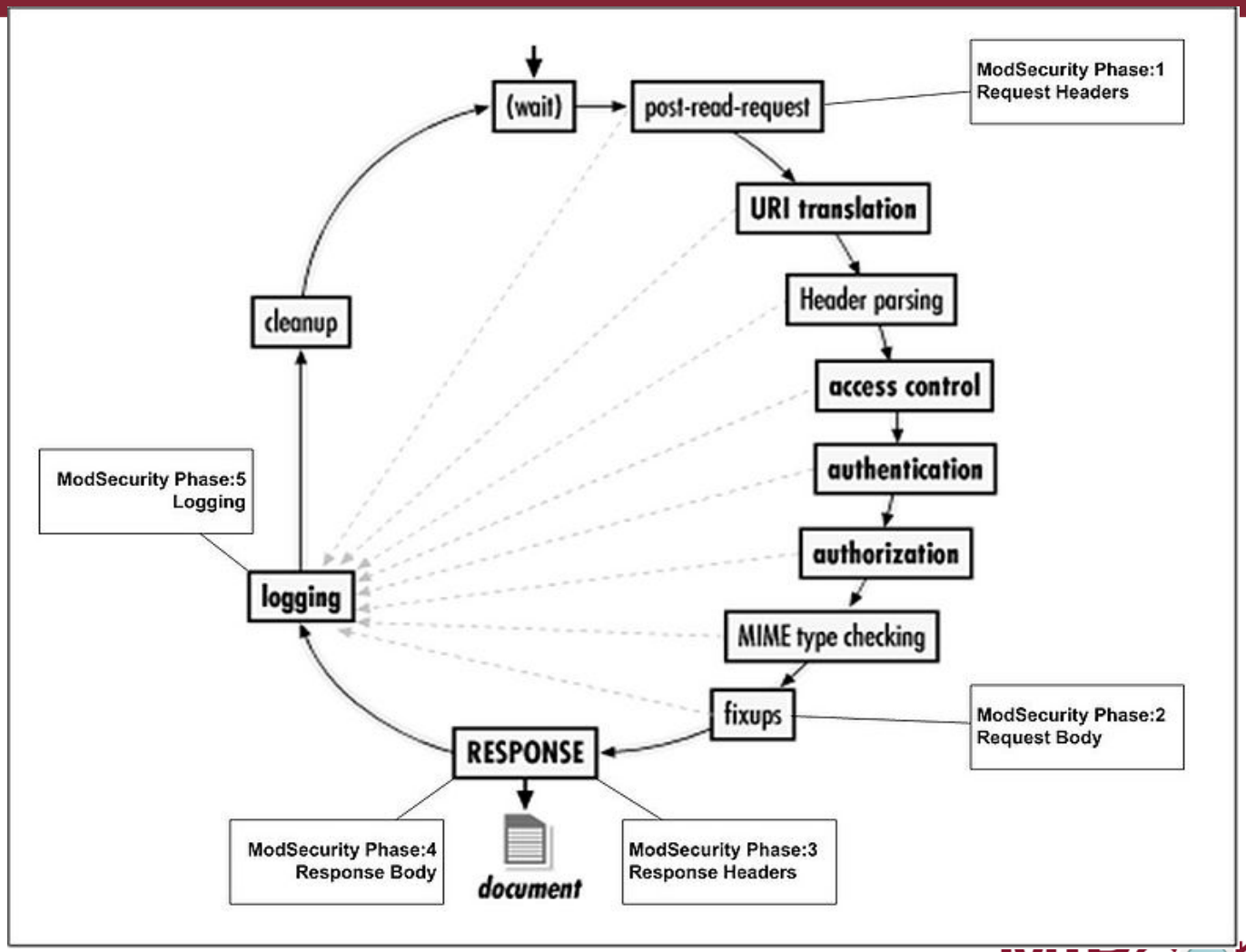
ModSecurity 101

- Web application firewall engine
- Runs on Apache, IIS7 and Nginx

- Traditional and Anomaly Scoring
- Inspect request and response data
- Greatly increased information logged

- OWASP ModSecurity Core Rule Set (CRS)
  - https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project
- Commercial rules also available from Trustwave

Processing Phases



http://tinyurl.com/ModSecPhases

Example OWASP ModSecurity CRS Rule

```
SecRule RESPONSE_BODY "<h2>Site
Error<\/h2>.{0,20}<p>An error was encountered while
publishing this resource\." \


"phase:4,rev:'2.2.0',t:none,capture,ctl:auditLogParts
=+E,block,msg:'Zope Information
Leakage',id:'970007',tag:'LEAKAGE/ERRORS_ZOPE',tag:'W
ASCTC/WASC-
13',tag:'OWASP_TOP_10/A6',tag:'PCI/6.5.6',severity:'3
',setvar:'tx.msg=%{rule.msg}',setvar:tx.outbound_anom
aly_score=+%{tx.error_anomaly_score},setvar:tx.anomal
y_score=+%{tx.error_anomaly_score},setvar:tx.%{rule.i
d}-LEAKAGE/ERRORS-%{matched_var_name}=%{tx.0}"
```

Traditional vs Anomaly Processing

- Traditional
  - Rule match -> Take the action configured in the rule
  - Easy to understand
  - No sharing of information between rules
  - SecDefaultAction "phase:2,deny,log"

- Anomaly
  - Rule match -> Increase an anomaly score
  - If anomaly score exceeds a threshold, then take an action
  - More difficult to understand
  - Provides increased flexibility in response
  - SecDefaultAction "phase:2,pass,log"

Audit Logging

- Two different logging types
  - Serial
  - Concurrent
- NOT line separated log files
  - Events are broken up into blocks of text
  - Block Separators
    - --<unique_hexidecimal_boundary>-<section_ID>--
    - --7d234671-H--

Log Section Definitions

- A – Audit log header
- B – Request headers
- C – Request body
- E – Intermediary response body
- F – Final response headers
- H – Audit log trailer
- J – Information about uploaded files
- K – Every rule matched by an event, in order of match
- Z – Final boundary

- https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual#wiki-SecAuditLogParts

Log Event Example

```
--bdacc45a-A--
[02/Oct/2013:18:07:24 --0600] Uky1PH8AAQEAABe6AhIAAAAC 10.10.1.1 49735 10.10.1.143 80
--bdacc45a-B--
POST /vulnerabilities/exec/ HTTP/1.1
Host: mircon
Connection: keep-alive
Content-Length: 40
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://mircon
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_5) AppleWebKit/537.36 (KHTML, like Gecko) Ch
rome/30.0.1599.66 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: http://mircon/vulnerabilities/exec/
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Cookie: security=low; PHPSESSID=9mpdpoq2ef7i03l10nf78dku03; security=low

--bdacc45a-C--
ip=%3B+cat+%2Fetc%2Fpasswd&submit=submit
--bdacc45a-F--
```

Avoid Logging Sensitive Data

- ModSecurity log tons of data, which could include….
  - Credit card numbers
  - Social Security numbers
  - Passwords
- Implement ModSecurity's data sanitization
- CRS optional rules

Audit Log Sanitization

- sanitiseArg – remove any data parameters by name
  - SectionAction “phase:5,nolog,pass,*sanitiseArg:password*”
- sanitiseRequestHeader – remove any HTTP headers by name
  - SectionAction “phase:5,nolog,pass, \ *sanitiseRequestHeader:Authorization*”
- sanitiseMatched – remove any data parameters whose name you don’t know yet
  - SecRule ARGS “@verifyCC \d{13,16}” “phase:5,nolog,pass, \ *sanitiseMatched*”
  - ModSecurity Handbook, page 69

Preparation and Configuration

- Harden the web server
- Configure ModSecurity
  - Use DetectionOnly
  - Increase log verbosity
  - Testing as a “good” user and a bad guy
  - Determine the processing mode
  - Tuning
  - Custom rules
- Going for bonus points
  - Implement honey traps
  - Track bad guys across the application

Identifying Attackers

- Active vs Passive Defense
  - Active defense doesn't necessarily mean hacking back
  - Configure ModSecurity to entice bad guys to identifying themselves
- Techniques for making bad guys stand out
  - HTML comments
    - <!--- shorten to /e/t/applogic to get error --->
  - robots.txt entries
    - /admin/remoteControl.aspx
  - Hidden parameters
  - Bogus cookies
- Use honey traps to take specific actions based on someone taking the bait

Custom Rules

- Why write custom rules?
  - Every web application is a bit different
  - Need to put in a patch while development works on a fix
  - Need to mask data that shouldn't be logged
  - Want to lay traps for the bad guys to fall into

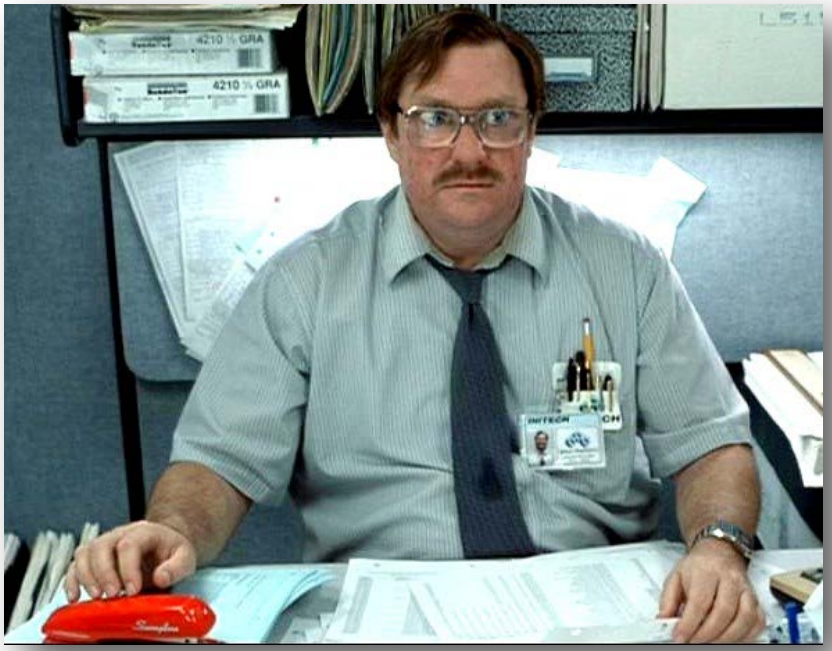
- Don't mix custom rules with distributed rules!

Example Custom Rule

```
SecRule
REQUEST_URI|REQUEST_BODY|REQUEST_HEADERS_NA
MES|REQUEST_HEADERS
"history.pushstate|history.replacestate"
"phase:4,deny,log,msg:'SH5ARK history-based
attacks detected'”
```

Analysis

- Manually analyzing the ModSecurity audit log may be hazardous to your mental health…



- Decide what to look for then find some tools fast!

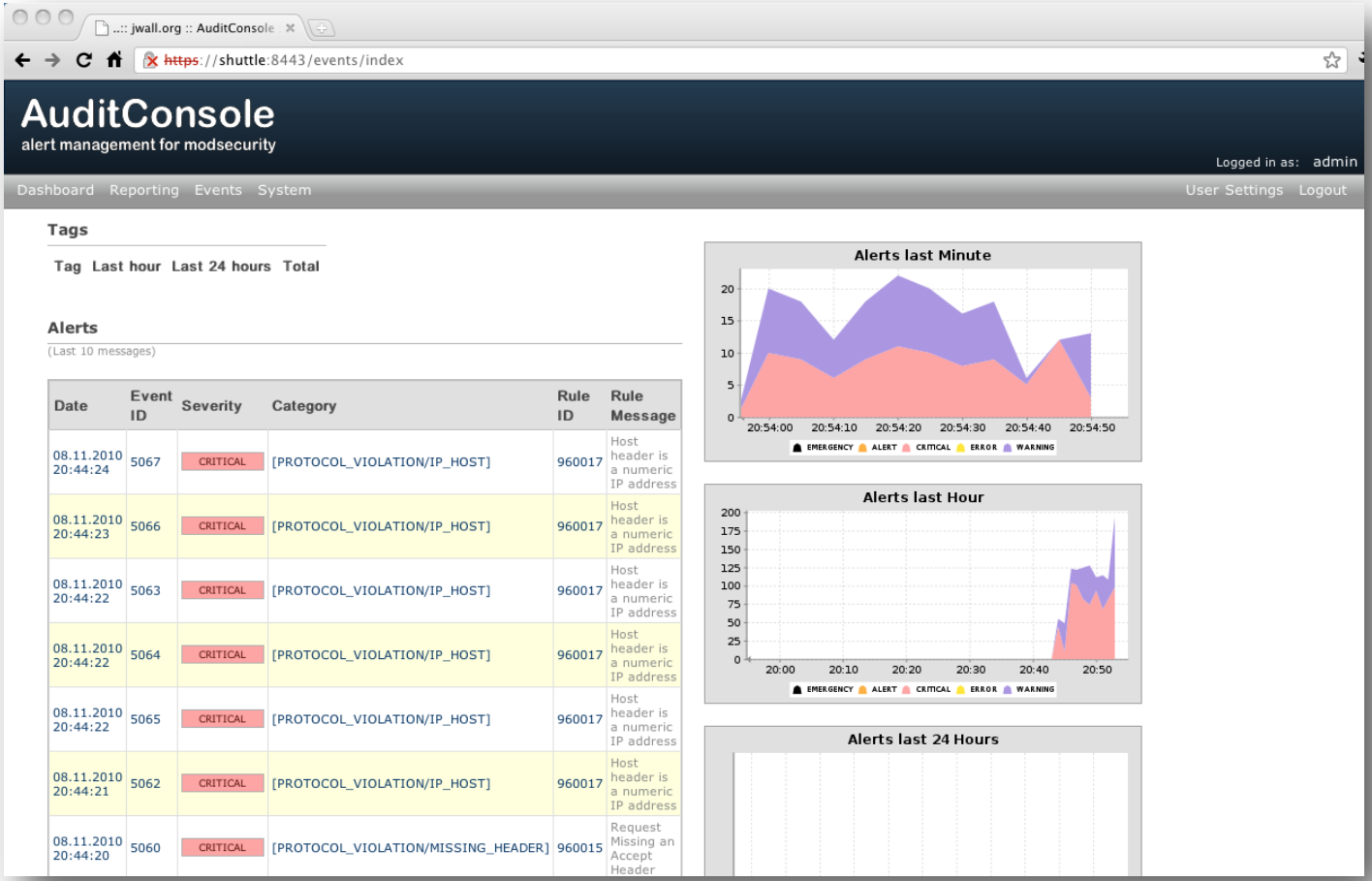
What Are We Looking For?

- Recent ModSecurity Alerts
  - SQL Injection
  - Cross Site Scripting (XSS)
  - Command Injection
  - Cookie Tampering, etc
- **Any** interaction with our honey traps
- Changes in the volume of application errors
- Changes in the volume of login attempts
- Application activity in odd time frames
- What data did an attacker send and what did he receive?

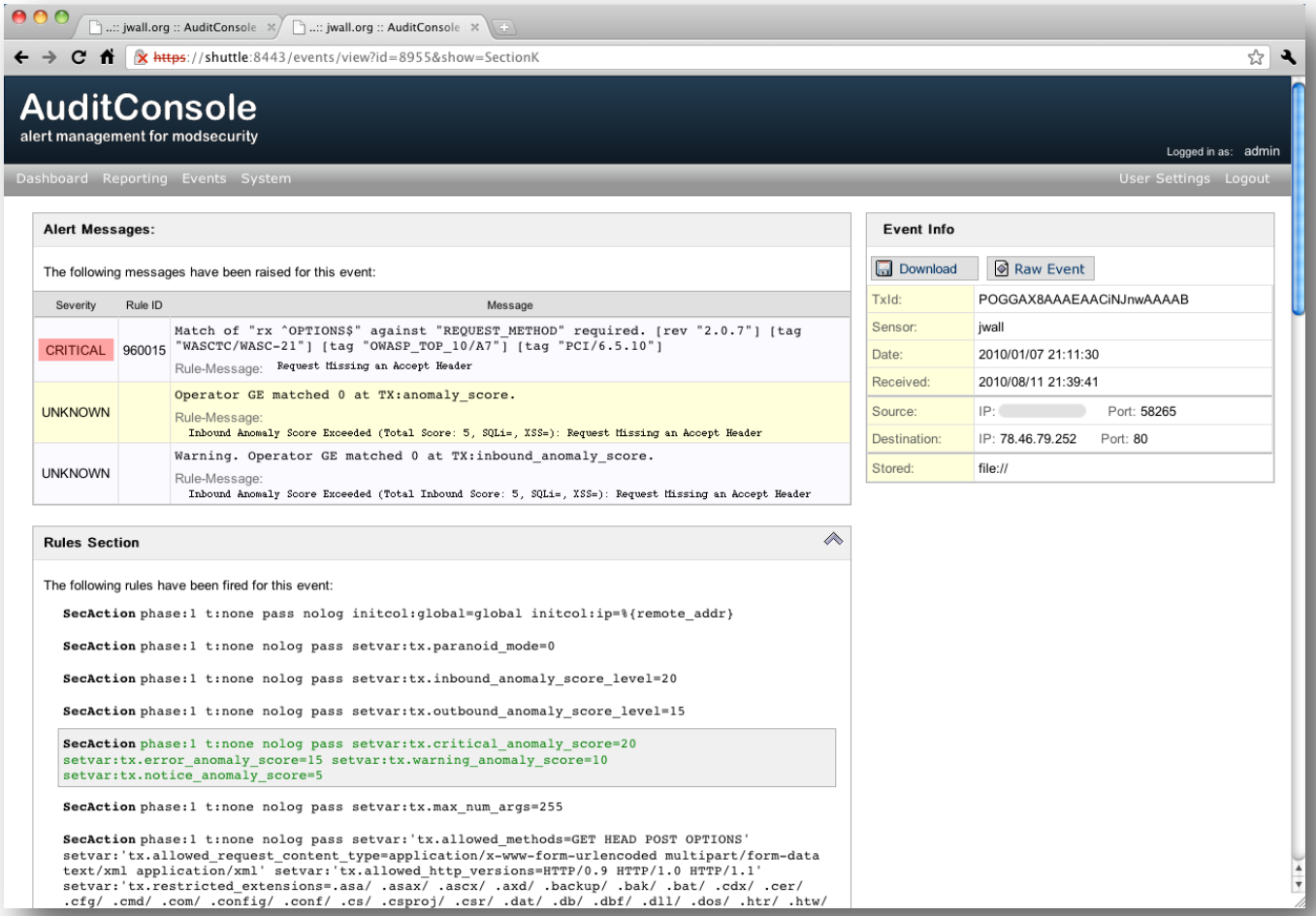
Audit Console

- Java application written for ModSecurity log analysis
- Creates a sensor which receives information via RPC interface
- Configure mlogc in Apache and ModSecurity to send events to the Audit Console listener

Audit Console Dashboard


..:: jwall.org :: AuditConsole
https://shuttle:8443/events/index
AuditConsole
alert management for modsecurity
Logged in as: admin
Dashboard Reporting Events System
User Settings Logout
Tags
Tag Last hour Last 24 hours Total
Alerts
(Last 10 messages)
Date Event ID Severity Category Rule ID Rule Message
08.11.2010 20:44:24 5067 CRITICAL [PROTOCOL_VIOLATION/IP_HOST] 960017 Host header is a numeric IP address
08.11.2010 20:44:23 5066 CRITICAL [PROTOCOL_VIOLATION/IP_HOST] 960017 Host header is a numeric IP address
08.11.2010 20:44:22 5063 CRITICAL [PROTOCOL_VIOLATION/IP_HOST] 960017 Host header is a numeric IP address
08.11.2010 20:44:22 5064 CRITICAL [PROTOCOL_VIOLATION/IP_HOST] 960017 Host header is a numeric IP address
08.11.2010 20:44:22 5065 CRITICAL [PROTOCOL_VIOLATION/IP_HOST] 960017 Host header is a numeric IP address
08.11.2010 20:44:21 5062 CRITICAL [PROTOCOL_VIOLATION/IP_HOST] 960017 Host header is a numeric IP address
08.11.2010 20:44:20 5060 CRITICAL [PROTOCOL_VIOLATION/MISSING_HEADER] 960015 Request Missing an Accept Header
Alerts last Minute
Alerts last Hour
Alerts last 24 Hours
EMERGENCY ALERT CRITICAL ERROR WARNING

Audit Console Event Viewer


..:: jwall.org :: AuditConsole
..:: jwall.org :: AuditConsole
https://shuttle:8443/events/view?id=8955&show=SectionK
AuditConsole
alert management for modsecurity
Logged in as: admin
Dashboard Reporting Events System
User Settings Logout
Alert Messages:
The following messages have been raised for this event:
Severity Rule ID Message
CRITICAL 960015 Match of "rx ^OPTIONS$" against "REQUEST_METHOD" required. [rev "2.0.7"] [tag "WASCTC/WASC-21"] [tag "OWASP_TOP_10/A7"] [tag "PCI/6.5.10"]
Rule-Message: Request Missing an Accept Header
UNKNOWN Operator GE matched 0 at TX:anomaly_score.
Rule-Message:
Inbound Anomaly Score Exceeded (Total Score: 5, SQLi=, XSS=): Request Missing an Accept Header
UNKNOWN Warning. Operator GE matched 0 at TX:inbound_anomaly_score.
Rule-Message:
Inbound Anomaly Score Exceeded (Total Inbound Score: 5, SQLi=, XSS=): Request Missing an Accept Header
Event Info
Download
Raw Event
TxId: POGGAX8AAAEAACiNJnwAAAAB
Sensor: jwall
Date: 2010/01/07 21:11:30
Received: 2010/08/11 21:39:41
Source: IP: Port: 58265
Destination: IP: 78.46.79.252 Port: 80
Stored: file://
Rules Section
The following rules have been fired for this event:
SecAction phase:1 t:none pass nolog initcol:global=global initcol:ip=%{remote_addr}
SecAction phase:1 t:none nolog pass setvar:tx.paranoid_mode=0
SecAction phase:1 t:none nolog pass setvar:tx.inbound_anomaly_score_level=20
SecAction phase:1 t:none nolog pass setvar:tx.outbound_anomaly_score_level=15
SecAction phase:1 t:none nolog pass setvar:tx.critical_anomaly_score=20 setvar:tx.error_anomaly_score=15 setvar:tx.warning_anomaly_score=10 setvar:tx.notice_anomaly_score=5
SecAction phase:1 t:none nolog pass setvar:tx.max_num_args=255
SecAction phase:1 t:none nolog pass setvar:'tx.allowed_methods=GET HEAD POST OPTIONS' setvar:'tx.allowed_request_content_type=application/x-www-form-urlencoded multipart/form-data text/xml application/xml' setvar:'tx.allowed_http_versions=HTTP/0.9 HTTP/1.0 HTTP/1.1' setvar:'tx.restricted_extensions=.asa/ .asax/ .ascx/ .axd/ .backup/ .bak/ .bat/ .cdx/ .cer/ .cfg/ .cmd/ .com/ .config/ .conf/ .cs/ .csproj/ .csr/ .dat/ .db/ .dbf/ .dll/ .dos/ .htr/ .htw/

Splunk

- Takes pretty much any data and makes it searchable
- Very flexible query language
- Easily combine your Apache and ModSecurity logs
- Build your own applications to monitor and search ModSecurity
  - http://kura2gurun.blogspot.com/2011/11/mod-security2-apps-for-splunk.html
- Regex for source type:
  - --[a-f0-9]+-[A|B|C|I|J|D|E|F|H|Z]--

Analyzing ModSecurity Data with Splunk

ModSecurity Resources

- Reference Manual
  - https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual
- Data Formats
  - https://github.com/SpiderLabs/ModSecurity/wiki/ModSecurity-2-Data-Formats
- Presentations
  - http://vimeo.com/search?q=ryan+barnett+modsecurity
- Books
  - ModSecurity Handbook
  - Web Application Defender's Cookbook

Questions?

Jason Wood

Email: jason@secureideas.com

Twitter: @Jason_Wood